

CEGO

1.0

Generated by Doxygen 1.8.14

Contents

1	README	1
2	Hierarchical Index	3
2.1	Class Hierarchy	3
3	Class Index	5
3.1	Class List	5
4	Class Documentation	7
4.1	CEGO::AbstractIndividual Class Reference	7
4.1.1	Detailed Description	8
4.1.2	Member Function Documentation	8
4.1.2.1	recombine_with()	8
4.2	CEGO::ALPSInputValues Struct Reference	8
4.2.1	Detailed Description	9
4.3	CEGO::ALPSReturnValues Struct Reference	9
4.3.1	Detailed Description	9
4.4	Antoine Class Reference	9
4.4.1	Detailed Description	10
4.5	CEGO::Bound Struct Reference	10
4.5.1	Detailed Description	10
4.5.2	Member Function Documentation	10
4.5.2.1	reflect_then_random_out_of_bounds()	10
4.6	Bumps Class Reference	11
4.6.1	Detailed Description	11

4.6.2	Member Function Documentation	11
4.6.2.1	f_givenxy() [1/2]	12
4.6.2.2	f_givenxy() [2/2]	12
4.7	BumpsInputs Struct Reference	12
4.7.1	Detailed Description	12
4.8	CEGO::numberish::id Union Reference	13
4.8.1	Detailed Description	13
4.9	CEGO::Layers< T > Class Template Reference	13
4.9.1	Detailed Description	14
4.9.2	Member Function Documentation	15
4.9.2.1	graduate_elderly_individuals()	15
4.9.2.2	print_diagnostics()	15
4.10	CEGO::numberish Struct Reference	15
4.10.1	Detailed Description	16
4.11	CEGO::NumericalIndividual< T > Class Template Reference	16
4.11.1	Detailed Description	17
4.11.2	Member Function Documentation	17
4.11.2.1	recombine_with()	17
4.12	RatPolyAncillary Class Reference	18
4.12.1	Detailed Description	18
4.13	CEGO::RecombinationFlags Struct Reference	18
4.13.1	Detailed Description	18
4.13.2	Member Data Documentation	19
4.13.2.1	nonuniform_w_stddev	19
4.13.2.2	p_same_w	19
4.13.2.3	uniform_w_stddev	19
4.14	CEGO::Result Struct Reference	19
4.14.1	Detailed Description	20
4.15	StornTest Class Reference	20
4.15.1	Detailed Description	20

Chapter 1

README

Eigen is a C++ template library for linear algebra: matrices, vectors, numerical solvers, and related algorithms.

For more information go to <http://eigen.tuxfamily.org/>.

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

CEGO::AbstractIndividual	7
CEGO::NumericalIndividual< T >	16
CEGO::ALPSInputValues	8
CEGO::ALPSReturnValues	9
Antoine	9
CEGO::Bound	10
Bumps	11
BumpsInputs	12
CEGO::numberish::id	13
CEGO::Layers< T >	13
CEGO::numberish	15
RatPolyAncillary	18
CEGO::RecombinationFlags	18
CEGO::Result	19
StornTest	20

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

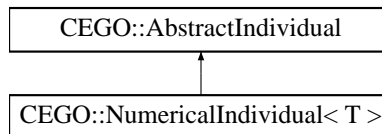
CEGO::AbstractIndividual	7
CEGO::ALPSInputValues	8
CEGO::ALPSReturnValues	9
Antoine	9
CEGO::Bound	10
Bumps	11
BumpsInputs	12
CEGO::numberish::id	13
CEGO::Layers< T >	13
CEGO::numberish	15
CEGO::NumericalIndividual< T >	16
RatPolyAncillary	18
CEGO::RecombinationFlags	18
CEGO::Result	19
StornTest	20

Chapter 4

Class Documentation

4.1 CEGO::AbstractIndividual Class Reference

Inheritance diagram for CEGO::AbstractIndividual:



Public Member Functions

- [AbstractIndividual](#) (std::size_t age)
True if the individual needs to have their objective function evaluated.
- void [request_evaluation](#) ()
Ask for an evaluation of this individual.
- void [set_needs_evaluation](#) (bool needs_evaluation)
Set the evaluation state of this individual.
- bool [needs_evaluation](#) () const
Returns true if evaluation is needed.
- void [increase_age](#) ()
Increase the age of the individual.
- std::size_t [age](#) () const
Return the age of the individual.
- void [set_age](#) (std::size_t age)
Set the age of the individual.
- virtual double [get_cost](#) ()=0
Return the cost.
- virtual void [calc_cost](#) ()=0
Calculate the cost of this individual.
- virtual Eigen::ArrayXd [get_coeffs_ArrayXd](#) ()=0
Get the coefficients as an array of doubles. Throws if not possible.
- virtual pIndividual [copy](#) () const =0
Return a copy of this individual.
- void [evaluate](#) ()
Evaluate the given individual; no-op if no evaluation is needed.
- virtual pIndividual [recombine_with](#) (pIndividual &other, const std::vector< [Bound](#) > &bounds, const [RecombinationFlags](#) &flags)=0
Merge this individual with another individual to obtain the offspring.

4.1.1 Detailed Description

Definition at line 261 of file datatypes.hpp.

4.1.2 Member Function Documentation

4.1.2.1 recombine_with()

```
virtual pIndividual CEGO::AbstractIndividual::recombine_with (
    pIndividual & other,
    const std::vector< Bound > & bounds,
    const RecombinationFlags & flags ) [pure virtual]
```

Merge this individual with another individual to obtain the offspring.

This must be implemented by the derived class; deciding how to handle crossover will depend on the data storage model (homogeneous or heterogeneous)

Implemented in [CEGO::NumericalIndividual< T >](#).

The documentation for this class was generated from the following file:

- include/CEGO/datatypes.hpp

4.2 CEGO::ALPSInputValues Struct Reference

Public Attributes

- std::vector< [Bound](#) > [bounds](#)
The vector of bounds on the variables.
- double [VTR](#)
Value to reach (terminates on reaching this cost value)
- CostFunction [f](#)
The cost function to be minimized.
- bool [parallel](#) = false
If true, evaluate each layer in a separate thread.
- std::size_t [max_gen](#) = 1000
Maximum number of generations that are allowed.
- std::size_t [NP](#) = 40
The number of individuals in a population (per layer)
- std::size_t [Nlayer](#) = 1
The number of layers.
- std::size_t [age_gap](#) = 5
The number of generations between restarting the bottom layer.
- bool [disp](#) = false
If true, display diagnostics as you go to standard out.

4.2.1 Detailed Description

Definition at line 634 of file CEGO.hpp.

The documentation for this struct was generated from the following file:

- include/CEGO/CEGO.hpp

4.3 CEGO::ALPSReturnValues Struct Reference

Public Attributes

- double [fval](#)
The function value at termination.
- double [elapsed_sec](#)
The number of seconds to conduct the entire optimization.
- std::string [termination_reason](#)
Why the optimization stopped.

4.3.1 Detailed Description

Definition at line 645 of file CEGO.hpp.

The documentation for this struct was generated from the following file:

- include/CEGO/CEGO.hpp

4.4 Antoine Class Reference

Public Member Functions

- **Antoine** (const std::string &name)
- void **plot_curve** ()
- Eigen::ArrayXd **eval_RHS** (const Eigen::ArrayXd &T, const Eigen::ArrayXd &c)
- double **objective** (const [CEGO::AbstractIndividual](#) *pind)
- double **objective** (const Eigen::ArrayXd &c)
- void **plot_trace** (const std::vector< double > &best_costs)
- const Eigen::ArrayXd & **get_T** ()

Public Attributes

- double **m_Tt**
- double **m_Tc**
- double **m_pc**
- double **m_Dc**
- Eigen::ArrayXd **m_LHS**
- Eigen::ArrayXd **m_T**
- std::string **m_name**

4.4.1 Detailed Description

Definition at line 21 of file Antoine.cxx.

The documentation for this class was generated from the following file:

- src/Antoine.cxx

4.5 CEGO::Bound Struct Reference

Public Member Functions

- `template<class T >`
Bound (const T &lower, const T &upper)
- `template<class T >`
Bound (const std::pair< T, T > &bounds)
- `template<typename URNG >`
void **gen_uniform** (URNG &gen, double &d, int &i) const
- **numberish enforce_bounds** (const **numberish** &n) const
- `template<typename URNG >`
numberish random_out_of_bounds (URNG &gen, const **numberish** &n) const
- `template<typename URNG >`
numberish reflect_then_random_out_of_bounds (URNG &gen, const **numberish** &val) const

Public Attributes

- **numberish m_lower**
- **numberish m_upper**

4.5.1 Detailed Description

Definition at line 126 of file datatypes.hpp.

4.5.2 Member Function Documentation

4.5.2.1 reflect_then_random_out_of_bounds()

```
template<typename URNG >
numberish CEGO::Bound::reflect_then_random_out_of_bounds (
    URNG & gen,
    const numberish & val ) const [inline]
```

< Excursion above the upper bound

< Excursion below the lower bound

< Width of the range

< Excursion above the upper bound

< Excursion below the lower bound

< Width of the range

Definition at line 177 of file datatypes.hpp.

The documentation for this struct was generated from the following file:

- include/CEGO/datatypes.hpp

4.6 Bumps Class Reference

Public Member Functions

- **Bumps** (std::size_t Nbumps, std::size_t Npoints)
- Eigen::ArrayXd **f_givenxy** (const Eigen::ArrayXd &xb, const Eigen::ArrayXd &yb, const Eigen::ArrayXd &x, const Eigen::ArrayXd &y)
Calculate the functional value for a set of vectors of points.
- double **objective** (const [CEGO::AbstractIndividual](#) *pind)
- double **objective_vec** (const std::vector< double > &c)
- double **penalty_vec** (const std::vector< double > &c)
- double **penalty** (const Eigen::ArrayXd &c)
- double **objective** (const Eigen::ArrayXd &c)
- void **plot_surface** ()
- void **plot_trace** (const std::vector< double > &best_costs)
- **Bumps** (std::size_t Nbumps, std::size_t Npoints, const std::vector< [CEGO::Bound](#) > &bounds)
- Eigen::ArrayXd **f_givenxy** (const Eigen::ArrayXd &c, const Eigen::ArrayXd &x, const Eigen::ArrayXd &y)
Calculate the functional value for a set of vectors of points.
- double **objective** (const [CEGO::AbstractIndividual](#) *pind)
- double **objective** (const Eigen::ArrayXd &cscaled)
- Eigen::ArrayXd **to_realworld** (const std::vector< [CEGO::numberish](#) > &x)
- Eigen::ArrayXd **to_realworld** (const Eigen::ArrayXd &x)
- Eigen::ArrayXd **to_scaled** (const Eigen::ArrayXd &x)
- void **plot_surface** ()
- void **plot_trace** (const std::vector< double > &best_costs)

Public Attributes

- std::size_t **Nbumps**
- Eigen::ArrayXd **xb0**
- Eigen::ArrayXd **yb0**
- Eigen::ArrayXd **xp**
- Eigen::ArrayXd **yp**
- Eigen::ArrayXd **zp**
- double **gamma** = 5
- Eigen::ArrayXd **c0**
- const std::vector< [CEGO::Bound](#) > **m_bounds**

4.6.1 Detailed Description

Definition at line 13 of file `inverse_gaussian.cxx`.

4.6.2 Member Function Documentation

4.6.2.1 f_givenxy() [1/2]

```
Eigen::ArrayXd Bumps::f_givenxy (
    const Eigen::ArrayXd & xb,
    const Eigen::ArrayXd & yb,
    const Eigen::ArrayXd & x,
    const Eigen::ArrayXd & y ) [inline]
```

Calculate the functional value for a set of vectors of points.

xb The x coordinate of the center of the bump yb The y coordinate of the center of the bump x The x coordinate of the points to be evaluated y The y coordinate of the points to be evaluated

Definition at line 38 of file inverse_gaussian.cxx.

4.6.2.2 f_givenxy() [2/2]

```
Eigen::ArrayXd Bumps::f_givenxy (
    const Eigen::ArrayXd & c,
    const Eigen::ArrayXd & x,
    const Eigen::ArrayXd & y ) [inline]
```

Calculate the functional value for a set of vectors of points.

xb The x coordinate of the center of the bump x The x coordinate of the points to be evaluated y The y coordinate of the points to be evaluated

Definition at line 57 of file shaped_inverse_gaussian.cxx.

The documentation for this class was generated from the following files:

- src/inverse_gaussian.cxx
- src/shaped_inverse_gaussian.cxx

4.7 BumpsInputs Struct Reference

Public Attributes

- std::string **root**
- std::size_t **parallel_threads**
- std::size_t **Nbumps**
- std::vector< std::size_t > **Nlayersvec**
- std::size_t **i**

4.7.1 Detailed Description

Definition at line 133 of file shaped_inverse_gaussian.cxx.

The documentation for this struct was generated from the following file:

- src/shaped_inverse_gaussian.cxx

4.8 CEGO::numberish::id Union Reference

Public Attributes

- double **d**
- int **i**

4.8.1 Detailed Description

Definition at line 17 of file datatypes.hpp.

The documentation for this union was generated from the following file:

- include/CEGO/datatypes.hpp

4.9 CEGO::Layers< T > Class Template Reference

Public Types

- typedef std::vector< std::tuple< std::size_t, plndividual > > **MutantVector**

Public Member Functions

- **Layers** (const std::function< double(const std::vector< T > &)> &function, std::size_t Nind_size, std::size_t Npop_size, std::size_t Nlayers, std::size_t age_gap=5)
- **Layers** (CostFunction &function, std::size_t Nind_size, std::size_t Npop_size, std::size_t Nlayers, std::size_t age_gap=5)
Constructor into which is passed a CostFunction and information about the layers.
- void **set_logging_scheme** (LoggingScheme scheme)
Specify the logging scheme that is to be employed.
- LoggingScheme **get_logging_scheme** ()
Get the logging scheme in use.
- void **set_filtering_function** (const std::function< FilterOptions(const **Result** &)> &f)
Set the filtering function that should be used.
- void **set_bounds** (const std::vector< **Bound** > &bounds)
Set the bounds on each element in the individual. If a one-element vector, the same bounds are used for each parameter.
- const std::vector< **Bound** > & **get_bounds** ()
Get the bounds applied to each element in the individual.
- const nlohmann::json **get_evolver_flags** () const
Get the flags for the evolver in JSON format.
- const void **set_evolver_flags** (const nlohmann::json &flags) const
Set the flags for the evolver in JSON format.
- void **set_builtin_evolver** (BuiltinEvolvers e)
Pick one of the builtin evolvers.
- const CostFunction & **get_cost_function** ()
Get the cost function that is being used currently.
- void **set_generation_mode** (GenerationOptions flag)

- *Set the flag to determine whether LHS or random (or other) is to be used to generate the population.*
- GenerationOptions [get_generation_mode](#) ()
- *Get the flag to determine whether LHS or random is to be used to generate the population.*
- void [graduate_elderly_individuals](#) ()
- void [repopulate_layers](#) ()
- *Repopulate the layer to replace removed old individuals.*
- void [evaluate_ind](#) (pIndividual &ind)
- *Evaluate a single individual, and store the values if needed.*
- void [evaluate_layers](#) ()
- *Evaluate all of the layers.*
- void [sort_all_layers](#) ()
- *Sort all of the layers.*
- void [increase_all_ages](#) ()
- *Increase the age of all individuals.*
- std::vector< std::map< std::string, double > > [cost_stats_each_layer](#) ()
- *Calculate statistics of the costs of individuals in each layer.*
- void [sort_layer](#) (Population &pop)
- *Sort a given layer.*
- void **parallel_evaluator** (MutantVector::iterator itstart, MutantVector::iterator itend, double &elap_sec)
- void **init_thread_pool** (short Nthreads)
- void **evaluate_mutants** (MutantVector &mutants)
- void **evolve_parallel** ()
- void [evolve_layer](#) (std::size_t i)
- *Evolve a given layer serially.*
- void [do_generation](#) ()
- *Carry out the steps for one generation.*
- std::vector< std::tuple< double, const std::vector< T > > > [get_best_per_layer](#) ()
- std::tuple< double, const std::vector< T > > [get_best](#) ()
- std::string [print_diagnostics](#) ()
- std::vector< [Result](#) > [get_results](#) ()
- *Get the results that have been logged during the course of this optimization.*

Public Attributes

- bool **parallel** = false
- bool **print_chunk_times** = false
- std::size_t **parallel_threads** = 6
- std::size_t **Nind_size**
- std::size_t **Npop_size**
- std::size_t **Nlayers**
- std::size_t **age_gap**

4.9.1 Detailed Description

```
template<typename T>
class CEGO::Layers< T >
```

Definition at line 130 of file CEGO.hpp.

4.9.2 Member Function Documentation

4.9.2.1 graduate_elderly_individuals()

```
template<typename T>
void CEGO::Layers< T >::graduate_elderly_individuals ( ) [inline]
```

Iterate over the layers and find individuals that are too old for the given layer

First try to see if the elderly individual dominates any individual in a layer with a higher age limit. If it does, replace the individual in the higher age limit layer.

Definition at line 247 of file CEGO.hpp.

4.9.2.2 print_diagnostics()

```
template<typename T>
std::string CEGO::Layers< T >::print_diagnostics ( ) [inline]
```

Return a string with some diagnostic information for the best individual in the population

See also

- get_best
- get_best_per_layer

Definition at line 618 of file CEGO.hpp.

The documentation for this class was generated from the following file:

- include/CEGO/CEGO.hpp

4.10 CEGO::numberish Struct Reference

Classes

- union [id](#)

Public Types

- enum **types** { **INT**, **DOUBLE** }

Public Member Functions

- **numberish** (const int &value)
- **numberish** (const double &value)
- void **operator=** (const int &value)
- void **operator=** (const double &value)
- **numberish operator-** (const **numberish** &value) const
- **numberish operator+** (const **numberish** &value) const
- **numberish operator*** (const **numberish** &value) const
- **operator int** () const
Get the value as an integer - stored double value will throw error.
- double **as_double** () const
Return the value as a double.
- int **as_int** () const
Return the value as an integer.
- **operator double** () const
Get the value as a double - stored integer will upcast to a double.
- std::string **to_string** () const
Convert the value to a string.

Public Attributes

- enum CEGO::numberish::types **type**
- union CEGO::numberish::id **u**

4.10.1 Detailed Description

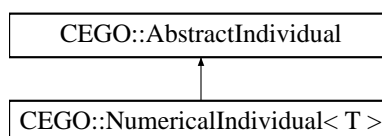
Definition at line 13 of file datatypes.hpp.

The documentation for this struct was generated from the following file:

- include/CEGO/datatypes.hpp

4.11 CEGO::NumericalIndividual< T > Class Template Reference

Inheritance diagram for CEGO::NumericalIndividual< T >:



Public Member Functions

- **NumericalIndividual** (const std::vector< T > &&c, const CostFunction &f)
- **NumericalIndividual** (const std::vector< T > &c, const CostFunction &f)
- const std::vector< T > & **get_coefficients** () const
- void **calc_cost** () override
Calculate the cost of this individual.
- void **set_cost** (T cost)
- double **get_cost** () override
Return the cost.
- Eigen::ArrayXd **get_coeffs_ArrayXd** ()
Get the coefficients as an array of doubles. Throws if not possible.
- virtual pIndividual **recombine_with** (pIndividual &other, const std::vector< Bound > &bounds, const RecombinationFlags &flags={}) override
Merge this individual with another individual to obtain the offspring.
- virtual pIndividual **copy** () const override
Return a copy of this individual.

4.11.1 Detailed Description

```
template<class T>
class CEGO::NumericalIndividual< T >
```

Definition at line 314 of file datatypes.hpp.

4.11.2 Member Function Documentation

4.11.2.1 recombine_with()

```
template<class T>
virtual pIndividual CEGO::NumericalIndividual< T >::recombine_with (
    pIndividual & other,
    const std::vector< Bound > & bounds,
    const RecombinationFlags & flags = {} ) [inline], [override], [virtual]
```

Merge this individual with another individual to obtain the offspring.

This must be implemented by the derived class; deciding how to handle crossover will depend on the data storage model (homogeneous or heterogeneous)

Implements [CEGO::AbstractIndividual](#).

Definition at line 339 of file datatypes.hpp.

The documentation for this class was generated from the following file:

- include/CEGO/datatypes.hpp

4.12 RatPolyAncillary Class Reference

Public Member Functions

- **RatPolyAncillary** (const std::string &name, const std::size_t Nnum, const std::size_t Nden)
- void **plot_curve** ()
- void **plot_deviation** (const Eigen::ArrayXd &c)
- Eigen::ArrayXd **eval_RHS** (const Eigen::ArrayXd &x, const Eigen::ArrayXd &c)
- double **objective** (const [CEGO::AbstractIndividual](#) *pind)
- double **objective** (const Eigen::ArrayXd &c)
- void **plot_trace** (const std::vector< double > &best_costs)

Public Attributes

- double **m_Tt**
- double **m_Tc**
- double **m_pc**
- double **m_Dc**
- std::size_t **m_Nnum**
- std::size_t **m_Nden**
- Eigen::ArrayXd **m_LHS**
- Eigen::ArrayXd **m_THETA**
- Eigen::ArrayXd **m_T**
- std::string **m_name**
- fitting_options **to_fit** = FIT_RHOV

4.12.1 Detailed Description

Definition at line 21 of file fit_ratpoly_ancillary.cxx.

The documentation for this class was generated from the following file:

- src/fit_ratpoly_ancillary.cxx

4.13 CEGO::RecombinationFlags Struct Reference

Public Attributes

- double **p_same_w** = 0.5
- double **nonuniform_w_stddev** = 1
- double **uniform_w_stddev** = 1
- bool **enforce_bounds** = true

4.13.1 Detailed Description

Definition at line 244 of file datatypes.hpp.

4.13.2 Member Data Documentation

4.13.2.1 nonuniform_w_stddev

```
double CEGO::RecombinationFlags::nonuniform_w_stddev = 1
```

The standard deviation for the normal distribution for the weighting factor between the individuals

Definition at line 248 of file datatypes.hpp.

4.13.2.2 p_same_w

```
double CEGO::RecombinationFlags::p_same_w = 0.5
```

The probability in [0,1] that the same w will be used to weight the entire individual in the recombination, otherwise different w will be used for each coefficient in the individual

Definition at line 245 of file datatypes.hpp.

4.13.2.3 uniform_w_stddev

```
double CEGO::RecombinationFlags::uniform_w_stddev = 1
```

The standard deviation for the normal distribution for the weighting factor between the individuals

Definition at line 250 of file datatypes.hpp.

The documentation for this struct was generated from the following file:

- include/CEGO/datatypes.hpp

4.14 CEGO::Result Struct Reference

Public Member Functions

- **Result** (Eigen::ArrayXd &&c, double &&ssq)

Public Attributes

- Eigen::ArrayXd **c**
- double **ssq**

4.14.1 Detailed Description

Definition at line 36 of file CEGO.hpp.

The documentation for this struct was generated from the following file:

- include/CEGO/CEGO.hpp

4.15 StornTest Class Reference

Public Member Functions

- **StornTest** (std::function< double(const std::vector< double > &)> &f, int NP, double F, double CR, double VTR, std::vector< [CEGO::Bound](#) > bounds)
- double **objective** (const [CEGO::AbstractIndividual](#) *pind)
- void **run** ()

Public Attributes

- int **Ncalls** = 0

Protected Attributes

- std::function< double(const std::vector< double > &)> **m_f**
- int **m_NP**
- double **m_F**
- double **m_CR**
- double **m_VTR**
- std::vector< [CEGO::Bound](#) > **m_bounds**

4.15.1 Detailed Description

Definition at line 55 of file StornPriceprofiling.cxx.

The documentation for this class was generated from the following file:

- src/StornPriceprofiling.cxx

Index

- Antoine, [9](#)
- Bumps, [11](#)
 - f_givenxy, [11](#), [12](#)
- BumpsInputs, [12](#)
- CEGO::ALPSInputValues, [8](#)
- CEGO::ALPSReturnValues, [9](#)
- CEGO::AbstractIndividual, [7](#)
 - recombine_with, [8](#)
- CEGO::Bound, [10](#)
 - reflect_then_random_out_of_bounds, [10](#)
- CEGO::Layers
 - graduate_elderly_individuals, [15](#)
 - print_diagnostics, [15](#)
- CEGO::Layers< T >, [13](#)
- CEGO::NumericalIndividual
 - recombine_with, [17](#)
- CEGO::NumericalIndividual< T >, [16](#)
- CEGO::RecombinationFlags, [18](#)
 - nonuniform_w_stddev, [19](#)
 - p_same_w, [19](#)
 - uniform_w_stddev, [19](#)
- CEGO::Result, [19](#)
- CEGO::numberish, [15](#)
- CEGO::numberish::id, [13](#)
- f_givenxy
 - Bumps, [11](#), [12](#)
- graduate_elderly_individuals
 - CEGO::Layers, [15](#)
- nonuniform_w_stddev
 - CEGO::RecombinationFlags, [19](#)
- p_same_w
 - CEGO::RecombinationFlags, [19](#)
- print_diagnostics
 - CEGO::Layers, [15](#)
- RatPolyAncillary, [18](#)
- recombine_with
 - CEGO::AbstractIndividual, [8](#)
 - CEGO::NumericalIndividual, [17](#)
- reflect_then_random_out_of_bounds
 - CEGO::Bound, [10](#)
- StornTest, [20](#)
- uniform_w_stddev
 - CEGO::RecombinationFlags, [19](#)